REMARKS/ARGUMENTS

Claims 1-11, 13, 15-18, 20, 21 and 23 are pending in the present application. Claims 1, 18, 21 and 23 were amended; and claim 12 was canceled. No claims were added. Support for the claim amendments can be found, for example, on page 26, lines 11-24; on page 27, line 27-page 28, line 13; and on page 48, lines 9-27. Reconsideration of the claims is respectfully requested in view of the above amendments and the following comments.

I. 35 U.S.C. § 101: Claims 21 and 23

The Examiner has rejected claims 21 and 23 under 35 U.S.C. § 101 as being directed towards non-statutory subject matter. This rejection is respectfully traversed.

In rejecting the claims, the Examiner states:

> **As to claim 21,** a "computer readable medium" is being cited, line 1, to include transmission-type media, light waves, radio frequency etc., cited in P. 48, lines 14-24, in the specifications; the claim is directed to a computer readable medium. However, Applicant defines "computer readable medium" to include "a computer data signal embodied in a carrier wave". Signals and carrier waves do not fall within any class of statutory subject matter, and thus the claim is not limited to statutory subject matter. Please see Interim Guidelines for Examination of Patent Applications for Patent Subject Matter Eligibility (1300 OG 142), Annex IV, Section (C) for details.
> **As to claim** 23, it is merely further recited as computer readable medium per se, thus, do not cure the deficiency of base claim 21, and also rejected under 35 U.S.C. 101 as set forth above.

Office Action dated October 18, 2007, pages 2-3.

In order to expedite prosecution, claim 21 has been amended to recite a computer program product including "a computer recordable-type medium storing computer readable program code for determining execution flow of a computer program. . . ." This language is supported in the present specification, for example, on page 48, lines 9-27, and excludes transmission-type media. Claim 21 and claim 23 dependent thereon, accordingly, now fully satisfy the requirements of 35 U.S.C. § 101 in all respects.

Therefore, the rejection of claims 21 and 23 under 35 U.S.C. § 101 has been overcome.

II. 35 U.S.C. § 103, Obviousness: Claims 1-13, 15-18, 20-21 and 23

The Examiner has rejected claims 1-13, 15-18, 20-21 and 23 under 35 U.S.C. § 103(a) as being unpatentable over Baba et al., US Published Patent Application No. US 2002/0010733 Al (hereinafter

"Baba"), in view of Hussain et al., US Patent No. 6,658,416 Bl (hereinafter "Hussain"). This rejection is respectfully traversed.

In rejecting the claims, the Examiner states:

> **As to claim 1** (Currently Amended), Baba discloses a method in a data processing system for autonomically determining execution flow of a computer program, comprising:
>
> - providing a set of hardware registers for identifying a work area for a thread of the computer program, wherein the work area stores thread tracking information for the thread (e.g., [0029] - Any registers in the processor may be assigned as the control reqister group. The stack machine rewrites the control register group as needed as it accesses the work area for the current thread according to the data in the register group; in this way it executes thread);
> - copying thread tracking information from the work area to a buffer using the set of hardware registers (e.g., [0003] - A stack machine consists of a memory area (or stack) where a work area is setup for each thread and a control device (hereafter referred to as a 'stack machine controller') which stacks work areas in this stack and executes each thread).
>
> Baba does not explicitly disclose retrieving symbolic data for the thread, wherein retrieving symbolic data for the tread includes retrieving symbolic data from an indexed symbolic database by searching the indexed symbolic database for symbolic data based on a process identifier for the thread; generating a call sequence of the computer program based on the symbolic data for the thread.
>
> However, in an analogous art of *Apparatus and Method for Creating an Indexed database of symbolic Data for Use with Trace Data of a Computer Program,* Hussain discloses:
>
> - retrieving symbolic data for the thread, wherein retrieving symbolic data for the tread includes retrieving symbolic data from an indexed symbolic database by searching the indexed 'symbolic database for symbolic data based on a process identifier for the thread (e.g., Fig. 10A, element of 1005 - Symbolic Data; Fig. 10B, element of 1070 - Symbolic Data; Fig. 13B, element of 1340; Fig. 18, elements 1830 - Search Indexed Database for Symbol Data Matching PID and Address, 1850 - Display Symbolic Data In Accordance With Trace File; Col. 16, Line 30 - Process identification (pid)); and
> - generating a call sequence of the computer program based on the symbolic data for the thread (e.g., Col. 2, Lines 38-41 - Such designers employ profiling tools to find characteristic code sequences and/or single instructions that require optimization for the available software for a given type of hardware).
>
> Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Hussain into the Baba's system to further provide retrieving symbolic data for the thread, wherein retrieving symbolic data for the tread includes retrieving symbolic data from an indexed symbolic database by searching the indexed symbolic database for symbolic data based on a process identifier for the thread; generating a call sequence of the computer program based on the symbolic data for the thread in Baba system.

> The motivation is that it would further enhance the Baba's system by taking, advancing and/or incorporating Hussain's system which offers significant advantages that store symbolic data for loaded modules during or shortly after a performance trace and utilizes the stored symbolic data when performing a performance analysis at later time; the merged symbol file contains information useful in performing symbolic resolution of address information in trace files for each instance of a module as once suggested by Hussain (e.g., Abstract).

Office Action dated October 18, 2007, pages 3-6.

Claim 1, as amended herein, is as follows:

> 1. A method in a data processing system for autonomically determining execution flow of a computer program, comprising:
>
> providing a set of hardware registers for identifying a work area for a thread of the computer program, wherein the work area stores thread tracking information for the thread;
>
> determining whether an overflow is about to occur in the work area;
>
> responsive to determining that an overflow is about to occur in the work area, copying the thread tracking information from the work area to a buffer using the set of hardware registers;
>
> retrieving symbolic data for the thread, wherein retrieving symbolic data for the thread includes retrieving symbolic data from an indexed symbolic database by searching the indexed symbolic database for symbolic data based on a process identifier for the thread; and
>
> generating a call sequence of the computer program based on the symbolic data for the thread.

The Examiner bears the burden of establishing a *prima facie* case of obviousness based on the prior art when rejecting claims under 35 U.S.C. §103. *In re Fritch*, 972 F.2d 1260, 23 U.S.P.Q.2d 1780 (Fed. Cir. 1992). Additionally, all limitations of the claimed invention must be considered when determining patentability. *In re Lowry*, 32 F.3d 1579, 1582, 32 U.S.P.Q.2d 1031, 1034 (Fed. Cir. 1994). Therefore, no *prima facie* obviousness rejection can be established if the proposed combination does not teach all of the features of the claimed invention. Furthermore, if an independent claim is nonobvious under 35 U.S.C. 103, then any claim depending therefrom is nonobvious. *In re Fine*, 837 F.2d 1071, 5 USPQ2d 1596 (Fed. Cir. 1988).

In the present case, no *prima facie* obviousness rejection can be stated against claims 1-11, 13, 15-18, 20, 21 and 23 because the combination of references, when considered as a whole, fails to teach or suggest all of the features of the claims. With respect to claim 1, for example, the proposed combination of references, when considered as a whole, does not teach or suggest at least the steps of "determining whether an overflow is about to occur in the work area", and "responsive to determining that an overflow is about to occur in the work area, copying the thread tracking information from the work area to a buffer using the set of hardware registers."

In rejecting the claims, the Examiner refers to paragraph [0003] of Baba as disclosing "copying thread tracking information from the work area to a buffer using the set of hardware registers", and refers to paragraph [0026] of Baba (in connection with the rejection of cancelled claim 12) as disclosing "copying thread tracking information from the work area to a buffer includes copying thread tracking information when a work area overflow occurs." Paragraphs [0003] and [0026] of Baba are reproduced below for the convenience of the Examiner:

> **[0003]** A computer with a control mechanism which can execute these Java byte codes (hereafter referred to as a "Java control mechanism") has a virtual machine called a "stack machine". This is used to execute a number of Java programs (or threads). A stack machine consists of a memory area (or stack) where a work area is set up for each thread and a control device (hereafter referred to as a "stack machine controller") which stacks work areas in this stack and executes each thread.
>
> **[0026]** According to one aspect of the invention, a data processing device contains a stack machine which secures a separate work area for each of a number of threads and executes the various threads while switching between them. When the control mechanism which activates this stack machine receives a request to switch stacks from the stack machine, a control circuit in the device controls the thread switching in response to this request.

The memory area referred to in paragraph [0003] above, is described as being an area where a work area is set up. Thus, the memory area and the work area in Baba are the same. Accordingly, paragraph [0003] of Baba does not disclose the buffer recited in claim 1, and does not disclose or suggest "copying the thread tracking information from the work area to a buffer", and also does not disclose or suggest "using the set of hardware registers" to perform such copying as recited in claim 1.

In addition, Baba also does not disclose using a set of hardware registers to perform such copying "responsive to determining that an overflow is about to occur in the work area" as recited in claim 1. Paragraph [0026] of Baba describes only that a separate work area is provided for each of a plurality of threads, and that a control circuit in the data processing device controls switching between the plurality of threads.

Neither in paragraph [0026] of Baba, nor anywhere else in Baba is there any disclosure or suggestion of "determining whether an overflow is about to occur in the work area", or "responsive to determining that an overflow is about to occur in the work area, copying the thread tracking information from the work area to a buffer using the set of hardware registers."

Hussain is cited as disclosing retrieving symbolic data for a thread and generating a call sequence based on the symbolic data. Hussain also does not disclose or suggest "determining whether an overflow is about to occur in the work area", or "responsive to determining that an overflow is about to occur in the work area, copying the thread tracking information from the work area to a buffer using the set of hardware registers", and, thus, does not supply these deficiencies in Baba.

For at least all the above reasons, neither Baba nor Hussain nor their combination discloses or suggests "determining whether an overflow is about to occur in the work area", or "responsive to determining that an overflow is about to occur in the work area, copying the thread tracking information from the work area to a buffer using the set of hardware registers" as recited in claim 1, and the Examiner has not established a *prima facie* case of obviousness in rejecting claim 1. Claim 1, accordingly, patentably distinguishes over the references in its present form.

Independent claims 18 and 21 have been amended in a similar manner as claim 1, and patentably distinguish over the cited art for similar reasons as discussed above with respect to claim 1.

Claims 2-11, 13, 15-17, 20 and 23 depend from and further restrict one of the independent claims and also patentably distinguish over the cited art, at least by virtue of their dependency. Furthermore, many of these claims recite additional features which are not disclosed or suggested by the cited art.

For example, claim 2 depends from claim 1 and recites that the set of hardware registers include "a work area register, a work area length register, and a current pointer register." Claim 4 depends from claim 2 and recites that the work area register includes "one of a size of the work area for the thread or a pointer pointing to an end of the work area for the thread"; and claim 5 depends from claim 2 and recites that the current pointer register includes "a pointer pointing to a location of the work area where last thread tracking information is written." In rejecting these claims, the Examiner refers to paragraph [0029] of Baba reproduced below, and to the register for stack head pointer, the register for program pointer and the register for local variable pointer, illustrated, for example, in Figures 4 and 6 of Baba:

> **[0029]** The stack machine controller is built into the aforesaid control mechanism. Any registers in the processor may be assigned as the control register group. The stack machine rewrites the control register group as needed as it accesses the work area for the current thread according to the data in the register group; in this way it executes the thread.

Although Baba may disclose various registers for accessing a work area, the reference does not disclose or suggest, either in paragraph [0029] of Baba or elsewhere, the specific registers recited in claims 2, 4 and 5. Claims 2, 4 and 5, accordingly, patentably distinguish over Baba in view of Hussain in their own right as well as by virtue of their dependency.

Furthermore, claim 13 depends from claim 1 and specifies that the buffer recited in claim 1 is one of a trace buffer and a consolidated buffer accessible by the application. Since, as discussed above, the references do not disclose a buffer to which tracking information is copied responsive to determining that an overflow is about to occur in a work area, the references also do not disclose or suggest that such a buffer may be a trace buffer or a consolidated buffer accessible by an application. Claim 13, accordingly, also patentably distinguishes over the cited art in its present form.

Therefore, the rejection of claims 1-13, 15-18, 20-21, and 23 under 35 U.S.C. § 103(a) has been overcome.

III. Conclusion

For at least all the above reasons, claims 1-11, 13, 15-18, 20, 21 and 23 patentably distinguish over the cited art and this application is believes to be in condition for allowance. It is, accordingly, respectfully requested that the Examiner so find and issue a Notice of Allowance in due course.

The Examiner is invited to call the undersigned at the below-listed telephone number if in the opinion of the Examiner such a telephone conference would expedite or aid the prosecution and examination of this application.

DATE: January 17, 2008

Respectfully submitted,

/Gerald H. Glanzman/
Gerald H. Glanzman
Reg. No. 25,035
Yee & Associates, P.C.
P.O. Box 802333
Dallas, TX 75380
(972) 385-8777
Attorney for Applicants